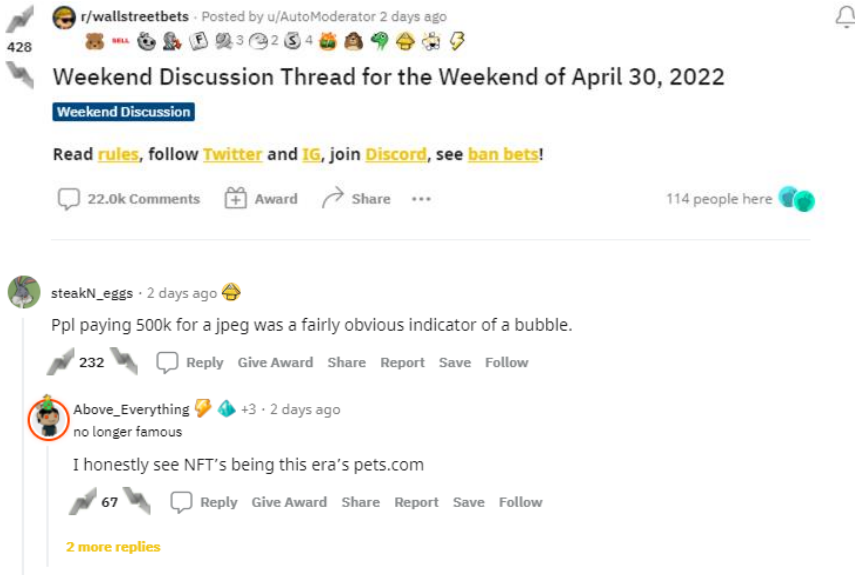
Reddit Graph Analysis

By Kyle Arbide

Data Description

- Data was collected from r/wallstreetbets
- Contains information from posts and comments on the subreddit
- Data was collected between 3/28/21 - 4/5/21
- Dataset contains:
  - 598 posts
  - 13552 comments
- Original data has a JSON-like format



Data Cleaning

There were a few key steps needed to clean the data:

- num_comments and num_upvotes fields need to be converted to int dtypes
- Datetime types not accepted in arango, need to be converted to stings
- Usernames are formatted differently for posts and comments, converted to display just the username for both


```
{'title': 'State of Reddit /r/Place 2022',
 'Reference_Date': datetime.datetime(2022, 4, 4, 19, 58, 36, 348707),
 'body': 'State of Reddit /r/Place 2022',
 'num_comments': '8.0k comments',
 'tag': 'News',
 'num_upvotes': '210',
 'author': 'u/AutoModerator',
 'comments': [{'comment_id': 't1_i3fd9sr',
   'username': '/user/SDAChess/',
   'body': ' Love from EPITA <3',
   'upvotes': '55'},
  {'comment_id': 't1_i3fd4y0',
   'username': '/user/WSBgirl/',
   'body': ' We did it!!!!! Great job everyone. Never thought I'd have such
trong opinions on Portugal, Baby Metal or Foxhole',
   'upvotes': '45'},
```

Deciding between JSON and Graph Database

JSON

Pros:

- Data is already in a JSON representation
- Simpler query language
- High performance (fast)

Cons:

- Difficult to understand the relationships between documents

Graph

Pros:

- Much easier to understand relationships between nodes
- Collection formatting requires a bit more work, but it is still relatively simple

Cons:

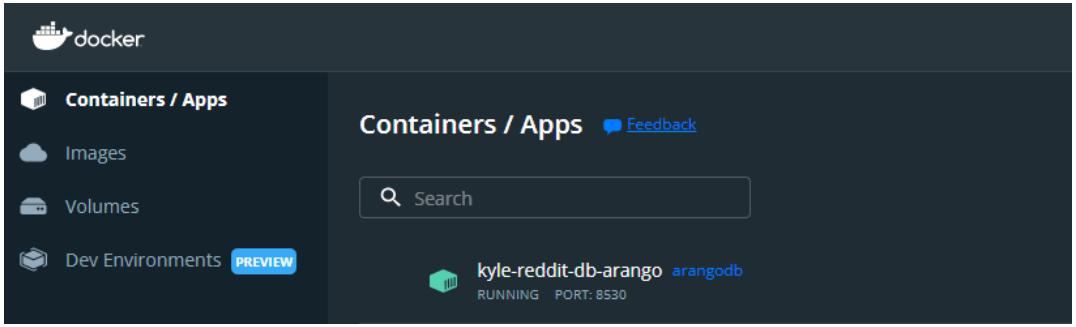
- AQL is slightly more complicated

Docker

I decided to use Docker to spin up my database. Some of the benefits of using Docker are:

- Can be easily created, spun-up, and shut-down
- Docker containers are portable, can easily be deployed on another system
- Docker APIs allow for quick and easy scaling

Unlike Virtual Machines, Docker containers are not accessible through the internet

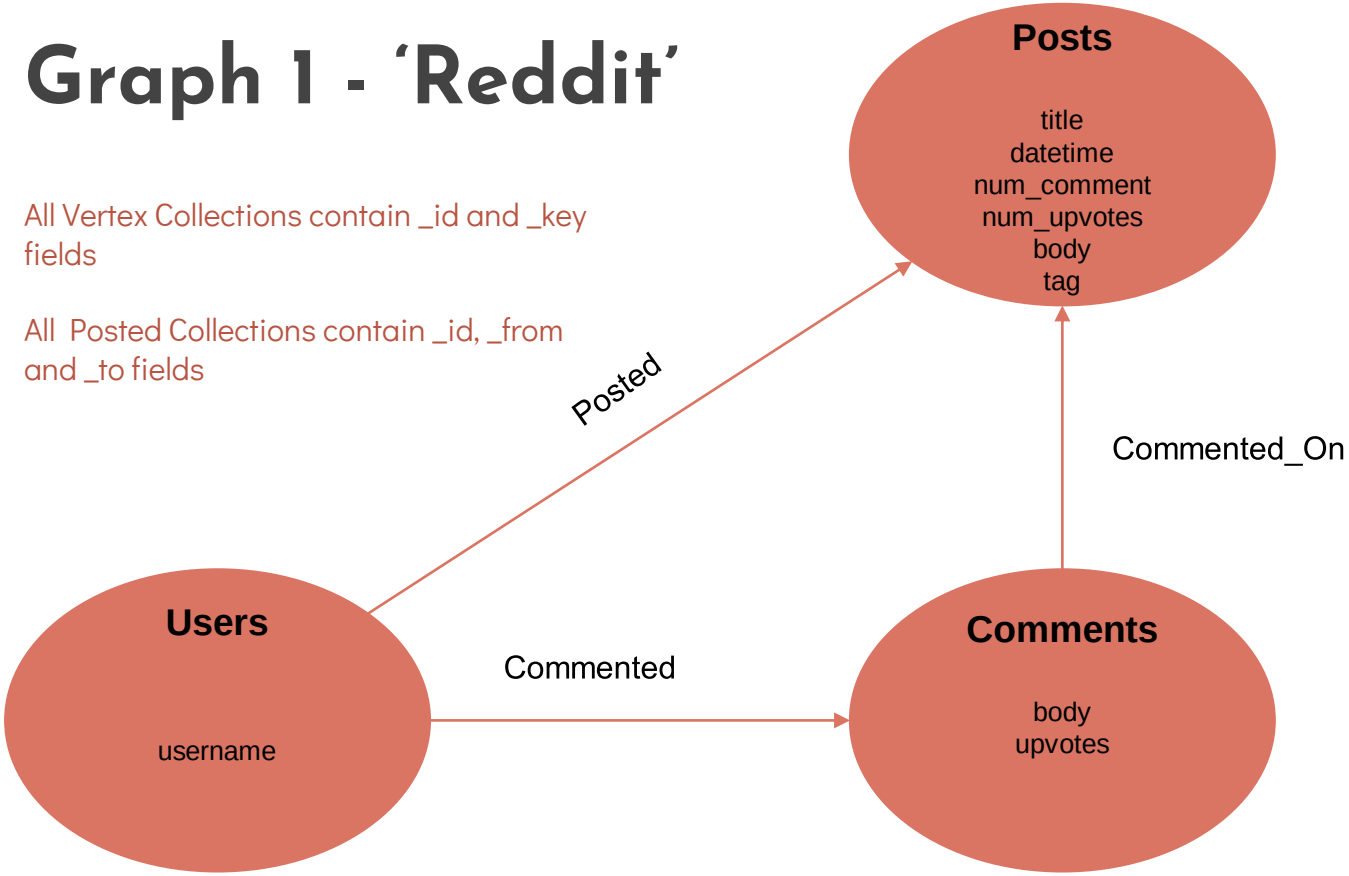


Schema

Graph 1 - 'Reddit'

All Vertex Collections contain _id and _key fields

All Posted Collections contain _id, _from and _to fields

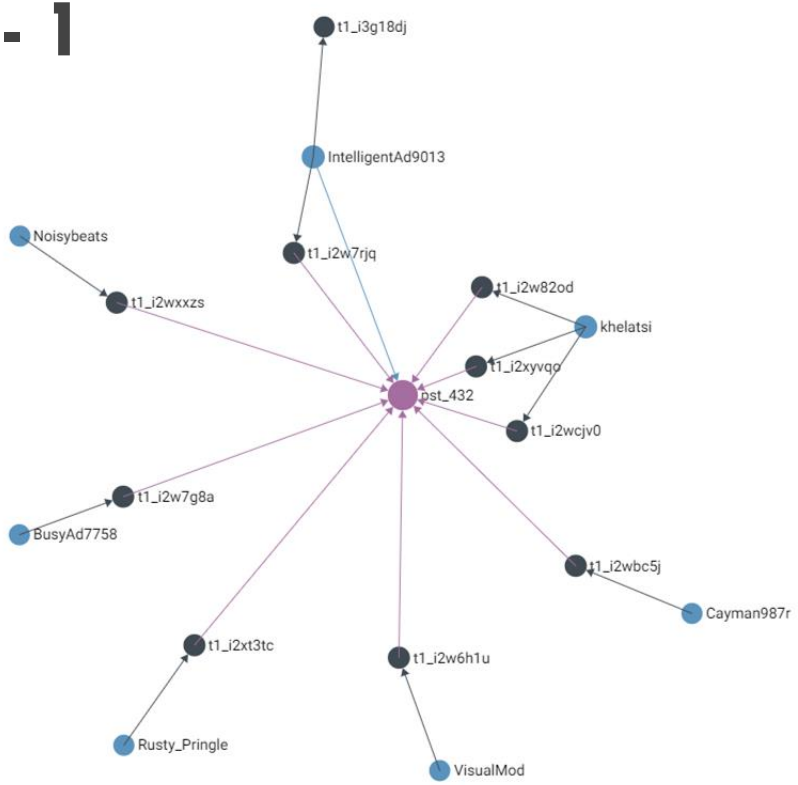


Arango Representation - 1

Legend:

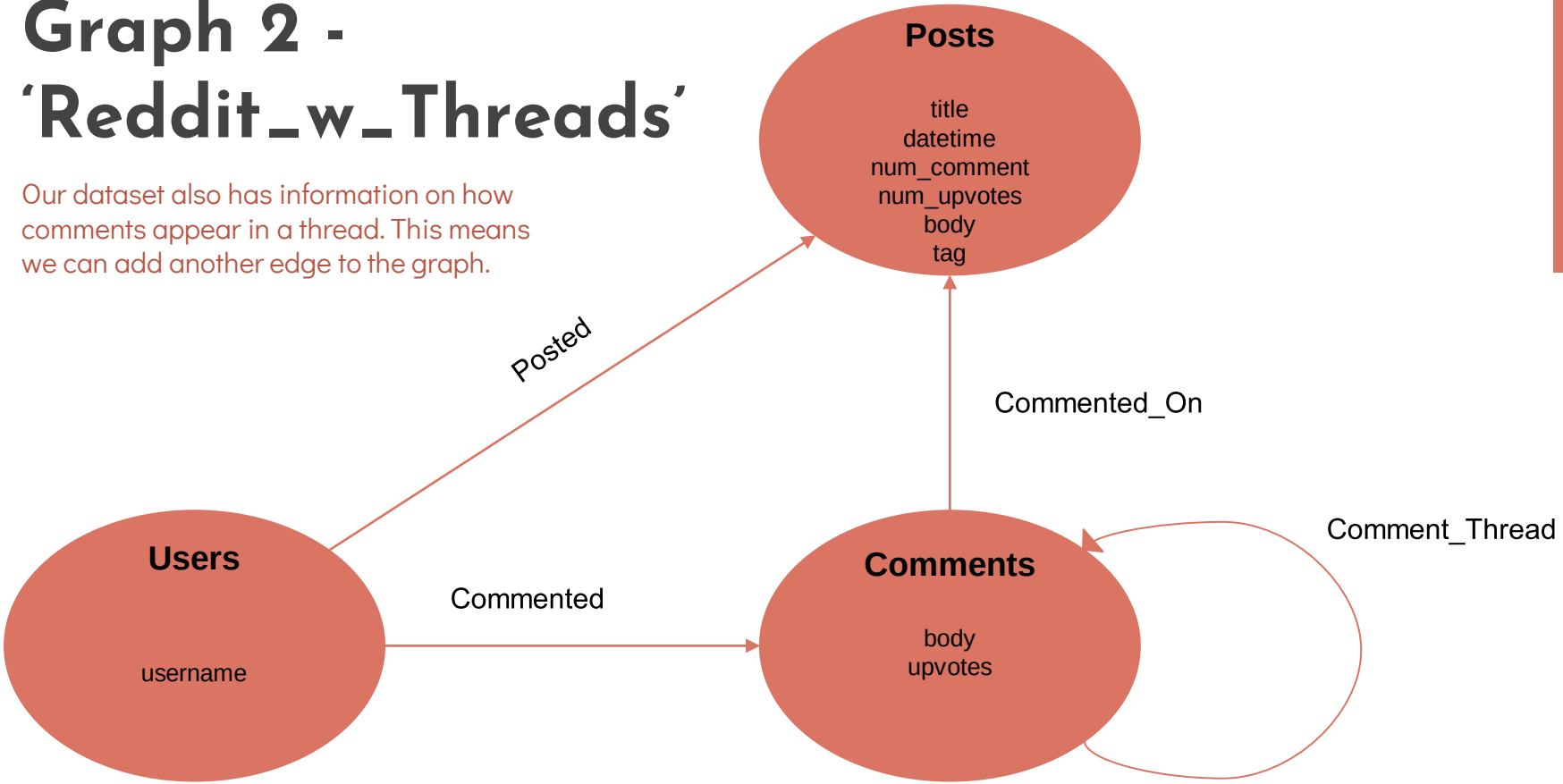
- Users — Posted
- Comments — Commented
- Posts — Commented_On

Depth: 2
Starting Node: pst_432



Graph 2 - 'Reddit_w_Threads'

Our dataset also has information on how comments appear in a thread. This means we can add another edge to the graph.

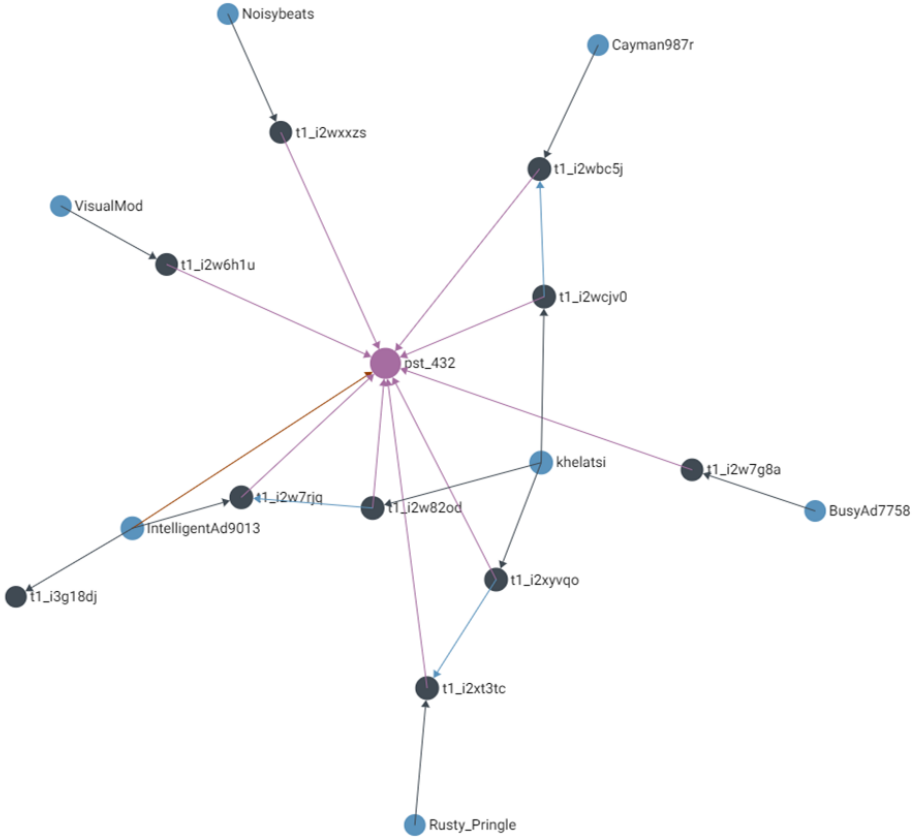


Arango Representation - 2

Legend:

Users	Posted
Comments	Commented
Posts	Commented_On
	Commented_Thread

Depth: 2
Starting Node: pst_432



Queries & Traversals

Queries

The following questions were answered using simple queries created with AQL:

Number of Posts with >300 Upvotes

```
post in posts
filter post.num_upvotes > 300
collect with count into upvotes
return upvotes
```

Returns: 170

Number of Comments with >300 Upvotes

```
for comment in comments
filter comment.upvotes > 300
collect with count into num_upvotes
return num_upvotes
```

Returns: 174

Number of Users with >5 Comments

```
for doc in commented
collect user = doc._from
aggregate num_comments =
        count_distinct(doc._to)
filter num_comments > 5
collect with count into num_users
return num_users
```

Returns: 339

More Queries!

Top 5 Upvoted Posts

```
for post in posts
sort post.num_upvotes desc
limit 5
return
{[post._id]:post.num_upvotes}
```

Returns:

{'posts/pst_176': 332},
{'posts/pst_444': 331},
{'posts/pst_445': 331},
{'posts/pst_446': 331},
{'posts/pst_447': 331}

Top 5 Users Based on Number of Comments

```
for doc in commented
    collect user = doc._from
    aggregate num_comments =
COUNT_DISTINCT(doc._to)
    sort num_comments desc
    limit 5
    return {[user]: num_comments}
```

Returns:

{'users/VisualMod': 414},
{'users/AutoModerator': 191},
{'users/Dfree707': 64},
{'users/MoneyManToTheMoon': 44},
{'users/limethedragon': 40}

Traversal 1 -Daily Discussion

Question: How far does the Discussion from Daily Discussion posts extend out into the subreddit?

Uses:

Graph 1 : 'Reddit'

Criteria:

Includes all users who commented on a daily discussion post.

If a commenter created a post, it includes all users who commented on their post.

Result:

117 Users

1.7% of Users (6,868 Total Users)

```
for v,e,p in 3..6 any 'users/AutoModerator' graph reddit
filter p.vertices[1].tag == 'Daily Discussion' &&
    (LENGTH(p.vertices) <= 4 || p.vertices[4].title)
filter v._id like 'users/%'
return distinct(v)
```

Traversal 2 -Topic Discussion

Question: How many users are participating in conversations about a given topic. In this case GameStop!

Uses:

Graph 2: 'Reddit_w_traversals'

Criteria:

Any user who posts or comments about GameStop
Any user who comments on a post about GameStop
Any user who comments in a comment thread about GameStop

Result:

1537 Users
22.38% of Users (6,868 Total Users)

If we had only included users that directly mentioned GameStop, it would have been 563 users (8.2%). Much less!

Pull the GME comments

```
for comment in comments
filter comment.body =~ "(?i)gme|gamestop"
return comment._id
```

Loop Through the comments and pull users from the thread

```
for v,e,p in 1..3 any '{comment}' graph reddit_w_threads
filter (p.edges[0]._id like "commented/%" && length(p.edges) == 1) ||
    (p.edges[0]._id like "comment_thread/%" &&
     p.edges[1]._id like "commented/%" &&
     not p.edges[2]._id) ||
    (p.edges[0]._id like "comment_thread/%" &&
    p.edges[1]._id like "comment_thread/%"  &&
    p.edges[2]._id like "commented/%")
filter v._id like 'users/%'
return distinct(v._key)
```

Conclusion

Using an Arango Database made it much easier to analyze the relationships between users, posts, and comments.

We have the choice between multiple graphs (with comment threads or without).

In the GameStop example, using extended relationships changed the results drastically.

Docker provided easy functionality to create, spin-up, and shut-down the database.

Gitlab: https://gitlab.com/c6831/2022_spring/kyle_reddit_graph_db/-/blob/main/raw_data_merge_and_prep.ipynb